January 9, 2002
Seminar Software Engineering

Title

ImmoWarrior

Report
Individual-Paper

Lucia Fimberger

lfimberg@cosy.sbg.ac.a

Academic Supervisor Mag. Dipl.-Ing. Schwaiger Roland

Department of Computer Science
University of Salzburg

Correspondence to:

Universität Salzburg
Institut für Computerwissenschaften und Systemanalyse
Jakob–Haringer–Straße 2
A–5020 Salzburg
Austria

ImmoWarrior

Lucia Fimberger
Department of Computer Science
University of Salzburg
A–5020 Salzburg, Austria
lfimberg@cosy.sbg.ac.at

Contents

Chapter 1

Introduction

1.1 Terms of Reference

A real-estate auction company ($e{-}Immo$) needs a new software for processing the acquisition of information and for the representation of this information by means of in support of electronic data processing ($ImmoWarrior$). These processes should be simple.
Potential users of the system are of course the collector of the information and the inquirer, who can be a person or a another system.
Resulting from this, there are two different scenarios:

- The collected information is passed on another system by ImmoWarrior. (XML)
- On condition that the collected inforamtion is passed on people by ImmoWarrior, there has to be an appropriate processing of the data.

A special requirement on the system is that ImmoWarrior should be easy to service, by meaning, it should not be necessary to install the components local.
Furtheron, $e - Immo$ should have the possibility to influence the data processing for individuals.

1.2 Project Development Progress

Points are listed in order of the homeworks.

1. CRC Cards
2. Use Cases
3. Essential User Interface Prototyping
4. Class Diagram
5. User Interface Flow Diagram and HTML and WML Prototypes (was group work)
6. Activity Diagram
7. Sequence Diagram

1.3 System Requirements

Following points have to be bear in mind:

- Collecting information (in our case real-estate).
- Possibility to enquire information.
- Real-estate collection.
- No local installation of components.
- User has to be identified.
- Different access rights.
- Processing of representation.
- Auction is online.
- User's administration.

1.4 Book

P. Stevens and R. Pooley:
UML Softwareentwicklung mit Objekten und Komponenten
Addison-Wesley

Chapter 2

CRC Cards

2.1 Basics on CRC Cards

CRC means Class-Responsibility-Collaboration. (invented by Beck and Cunningham 1989).
Although CRC cards are not part of UML, they are used in an early stage of software development to identify classes and their assoziations. CRC Cards are a technique for helping create a better object model for an object-oriented system.

2.1.1 Creating CRC Cards

A CRC card contains:

Name of the class (at the top)

Responsibilities of the class (on the left side)

Collaborators of the class, which support the execution of the responsibilities (on the right side)

The general idea is looking through the requirements for all the nouns that play a part in the functionality. These nouns become the classes.
Responsibilities identify problems to be solved. A responsibility serves as a handle for discussing potential solutions. The responsibilities of an object are expressed by a handful of short verb phrases, each containing an active verb. The more that can be expressed by these phrases, the more powerful and concise is the design. One class, normally, should not has more than three or four responsibilities. Are there to much responsibilities, it should be thought of subdividing the class.
One of the distinguishing features of object design is that all objects stand in relationship to others, on whom they rely for services and control. At least, for characterizing object designs: the collaborators of an object. Collaborators are objects which will send or be sent messages in the course of satisfying responsibilities. Collaboration is not necessarily a symmetric relation.

Finding Classes Read through the requirements and use cases looking for nouns and noun phrases. Make a list of these, changing the plurals to singulars, then throw out all words and phrases that are obviously nonsense. While doing this, convert all the passive sentences to active sentences. Also, convert all pronouns (I, it, he, they, etc.) to the noun that they stand for.

Be careful with adjectives - they may be irrelevant, or they may indicate different behaviors, and therefore potentially different subclasses.

Finding Responsibilities Responsibilities of an object include both the knowledge it maintains and the actions it can perform. When considering responsibilities, remember that they are the publicly available services, and that they specify what the object does, not how it does it.

Finding Collaborations By assigning responsibilities, consider if the class is capable of fulfilling the responsibility by itself. If not, what help does it need from other classes? These other classes will become collaborators.

2.1.2 CRC Session

The members of a CRC session are a project leader, facilitator and some colloborators. The goal of a CRC session is the discussion of the problems of the system. Furtheron, it is a quite good possibility to increase the team spirit.

2.2 CRC Cards for ImmoWarrior

This is the first very rough draft of the CRC Cards:

Real-Estate	
knows its characteristics (kind, objectname, size, address, age, status, ...	

Client	
knows its characteristics (name, address, ...)	
can search Real-Estates	Search

Auction	
knows Real-Estates	Real-Estates
handels bid	

Searching	
searches Real-Estates	Real-Estates
returns results	

Input Data	
handels input of Real-Estates	Real-Estates
handels input of Clients	Client

Seller	
knows his Real-Estates	Real-Estates
can input data	Input data

Second draft:

Real-Esate	
knows its characteristics	

Collection of Real-Estates	
knows Real-Estates	Real-Estates

User	
knows its characteristics	

Collection of Users	
knows Users	User

Enquirer	
recieves searching criteria	Data Converter
searches	Collection of Real-Estate
transmits results	Data Converter

Data Converter	
recieves and transmits data	Terminal, Enquirer
formats data	

Terminal	
recievs and displays data	Data Converter

Recorder	
recieves data	Data Converter
can input data	Collection of Real-Estates

Chapter 3

Use Case

3.1 Basics of Use Cases

Use Cases are a critical technique in developing an application. Within the UML Use Cases are used primarily to capture the high level user-functional requirements of a system. This long winded description is important because Use Cases cannot usefully be used to capture non-functional requirements. Nor can they usefully be used to capture ”internal” functional requirements. Attempting either or both is a sure path to disaster for two reasons. Firstly because Use Cases are an informal and imprecise modelling technique. But then they were never intended to be anything else. Secondly because the other use that we make of Use Cases is to define the fundamental structure of our application. The Use Case is not only important as a unit of requirement definition but also as our unit of estimation and our unit of work.
The Use Case model is about describing WHAT our system will do at a high-level and with a user focus for the purpose of scoping the project and giving the application some structure. The Use Cases are the unit of estimation and also the smallest unit of delivery. Each increment that is planned and delivered is described in terms of the Use Cases that will be delivered in that increment.
Use Cases are not a functional decomposition model. Use Cases are not intended to capture all of the system requirements. Use Cases do not capture HOW the system will do anything - nor do they capture anything the actor does that does not involve the system. All of these things are better modelled using other modelling techniques that were developed for those purposes. The Object Model to capture the static structure of the system and the composition of the classes. Object Sequence Diagrams and State Transition Diagrams to capture the detailed dynamic behaviour of the system - the HOW. The Business Process Model to capture the overall business processes - both computerised and manual.
Use Cases are not an inherently object-oriented modelling technique. There is no fundamental reason why they couldn’t be used as the front-end to a structured development method - but they’re not because the methods gurus are concentrating on the development of OO methods.

Elements of Use Case An actor, who is described by a matchstick man, stands for the user of the system. The actor is connected to an use case (by a line) if he is allowed to interact with the system to assist by the execution of the jobs. Pay attention to two things:

1. The actor is not an individual person, but a role, which someone is playing.

2. The communication relation between the actor and an Use Case do not mean, that someone, in that role, assists realla by the execution of the jobs. It means actually, that the possibility is given.

3.2 Use Cases of ImmoWarrior

The following points describe Fig. 3.1.

3.2.1 Record

Description User can input a new real-estate.

Precondition

Postcondition A new real-estate is stored in the datacollection.

	user	system
1.		presents the input mask
2.	inputs data of real-estate and submits	
3.		verifies data and ask for a recheck
4.	okay	stores data
	cancel	rejects data, goes to point 1

3.2.2 Enquiry Real-Estates

Description User can search real-estates with help of different searching criteria.

Precondition

Postcondition User gets the searching results.

	user	system
1.		presents enquiry mask
2.	chooses different serching criteria and submits	
3.		searches real-estates refering to searching criteria and presents results to user

3.2.3 Participate an Auction

Description User can bid.

Precondition User has found his favorite real-estate.

Postcondition User holds highest bid.

	user	system
1.		presents details of choosen real-estate
2.	bids	
3.		ask to submit
4.	okay	actualizes data of real-estate
	cancel	back to point 1

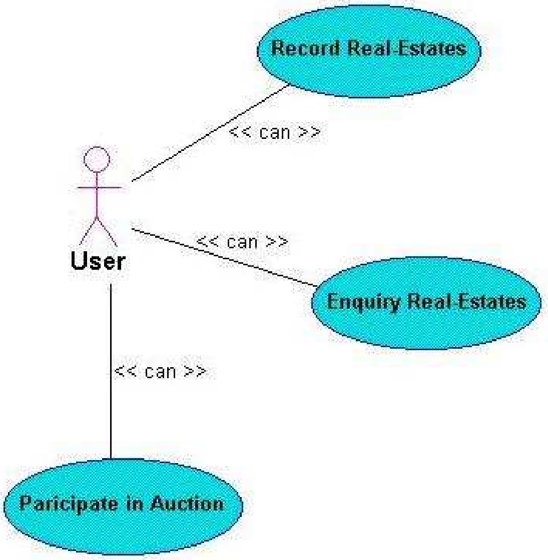


Figure 3.1: Use Case Diagram of ImmoWarrior

Chapter 4

Essential User Interface Prototyping

This is only a rough draft:

Start

Buttons: News Search for Real-Esates Go to Auction Contact	Text: Welcome and a short description of the company

News

The latest inputed Real-Estates are displayed.

Search

Text:	Howto's for searching
Criteria	Kategory, size, location, prize, age, number of rooms, ...

Searching Results

Displays searching results
Buttons: For Details
Buttons: Back

Details

Displays Details of required Real-Estate
Buttons: Back

Contact

For sending an E-Mail.
Subjectm, text, sender

Record Real-Estate

Edit fields: Of real-estate attributes (Name, Size, Prize,...)
Buttons: Ok, Back

Chapter 5

Different Diagrams

5.1 Class Diagram

5.1.1 Basics of Class Diagrams

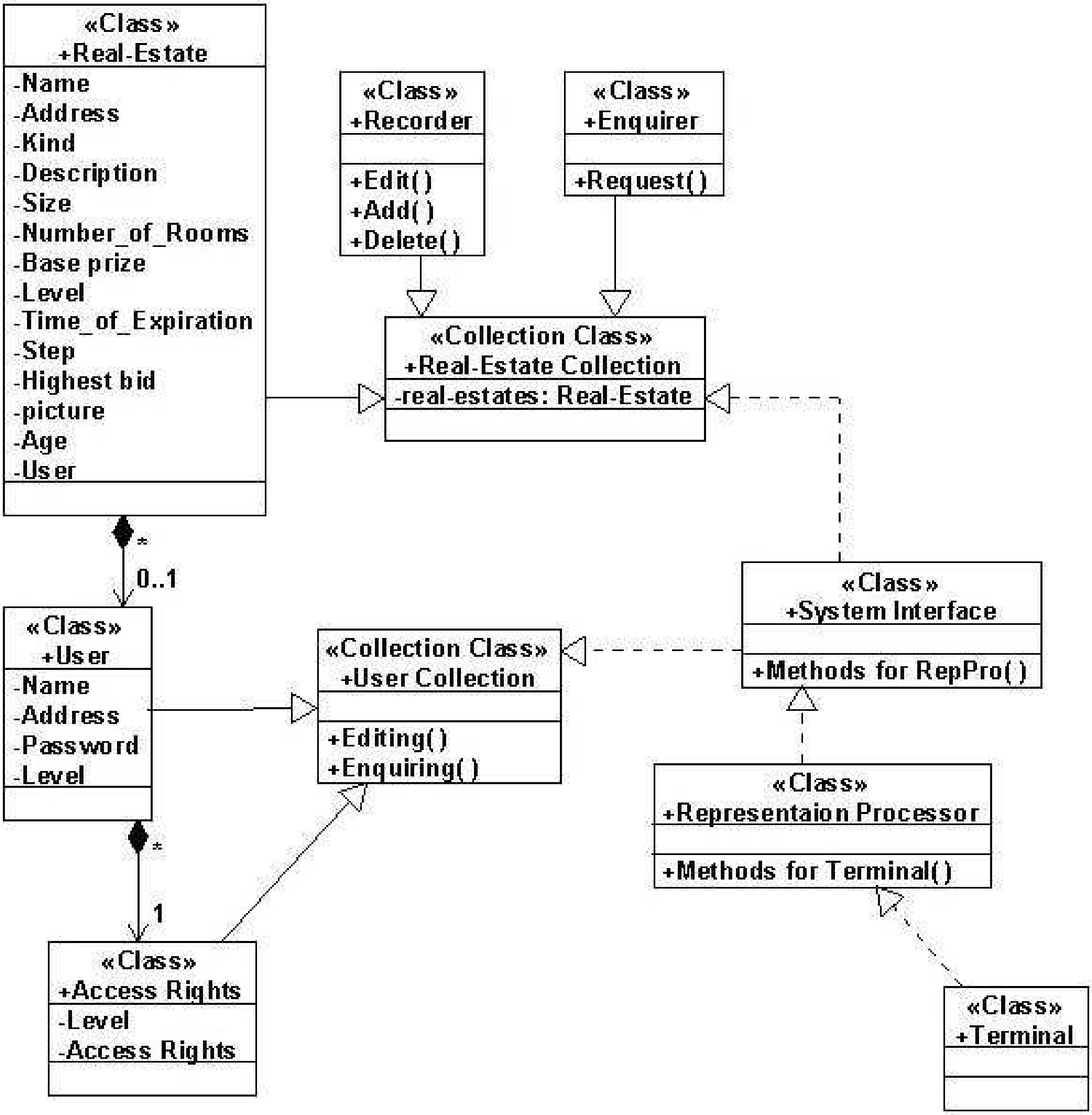
The class diagram is a central modeling technique that runs through nearly all object-oriented methods. This diagram describes the types of objects in the system and various kinds of static relationships which exist between them. There are three principal kinds of relationships which are important: associations, subtypes and aggregation.

Associations represent relationships between instances of types. The interpretation of them varies with the perspective. Conceptually they represent conceptual relationships between the types involved. In specification these are responsibilities for knowing, and will be made explicit by access and update operations.

Associations may be bi-directional (can be navigated in either direction) or uni-directional (can be navigated in one direction only). Conceptually all associations can be thought of as bi-directional, but uni-directional associations are important for specification and implementation models. For specification models bi-directional associations give more flexibility in navigation but incur greater coupling. In implementation models a bi-directional association implies coupled sets of pointers, which many designers find difficult to deal with.

One of the key aspects of associations is the cardinality of an association (sometimes called multiplicity). This specifies how many companies a person may work for, how many children a mother can have, etc. This corresponds to the notion of mandatory, optional, 1-many, many-many relationships in the Entity-Relationship approach. The UML has standard the mass of different approaches that existed here.

5.1.2 Class Diagram



5.2 Activity Diagram

5.2.1 Basics of Activity Diagrams

The activity diagram focuses on activities, chunks of process that may or may not correspond to methods or member functions, and the sequencing of these activities. In this sense it is like a flow chart. It differs, however, from a flow chart in that it explicitly supports parallel activities and their synchronization.

What is an Activity Diagram:

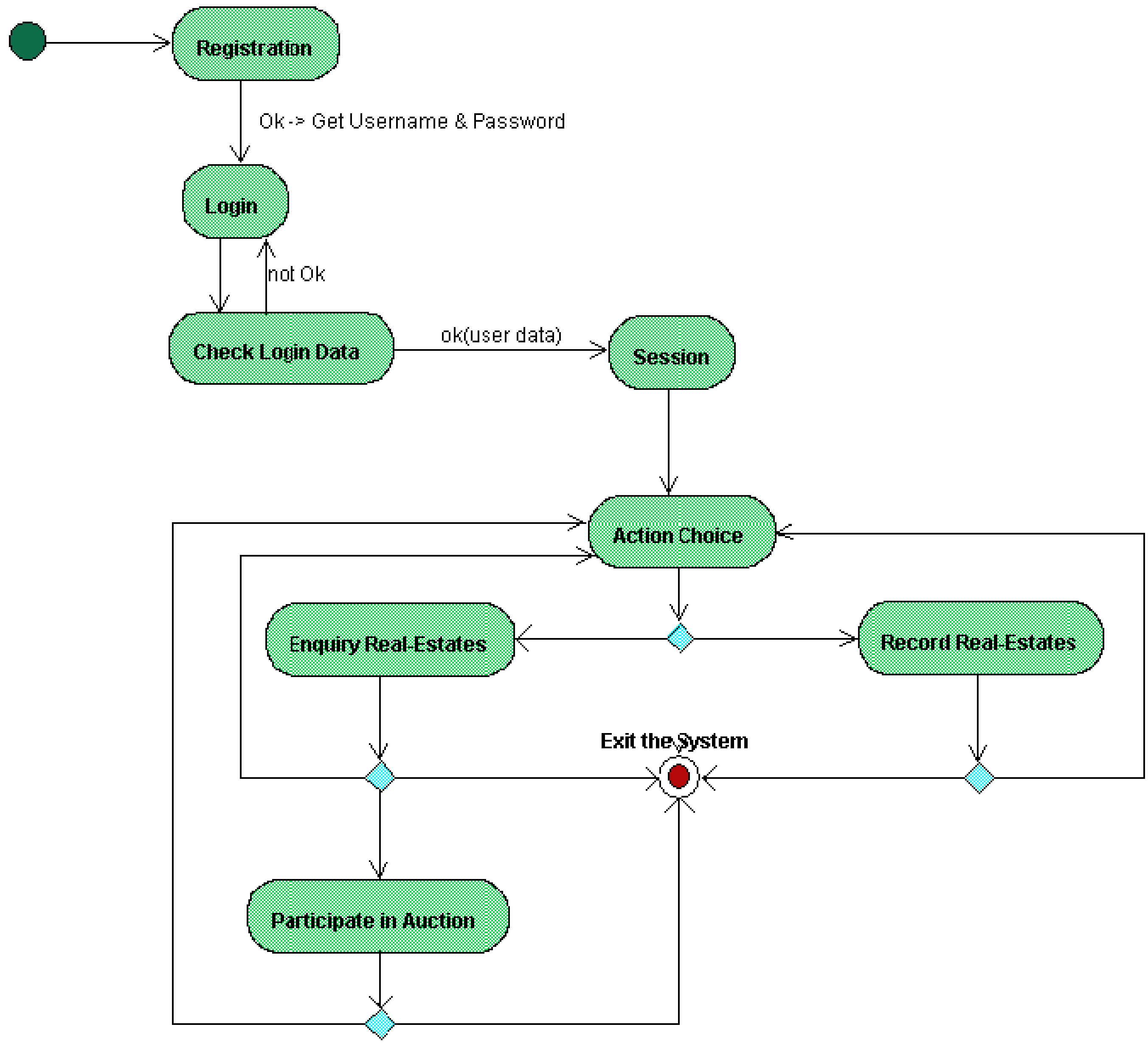
- Describes activities and flows of data or decisions between activities.
- Provides a very broad view of business processes.
- Can be used to break out the activities that occur within a use case.

When to use an Activity Diagram:

- When describing work flow across many use cases.
- When analyzing a use case.

This diagram is useful in showing work flow connections and describing behavior that has a lot of parallel processing.

5.2.2 Activity Diagram



5.3 Sequence Diagram

5.3.1 Basics of Sequence Diagrams

Object Sequence Diagrams are about deciding and modelling HOW our system will achieve WHAT we described in the Use Case model. During analysis this will still be at quite a high level possibly without even any user interaction modelled. But into design the OSD will be a complete description of exactly how each object in our system interact in order to perform a particular task. That task equates to a Use Case.

5.3.2 Sequence Diagrams

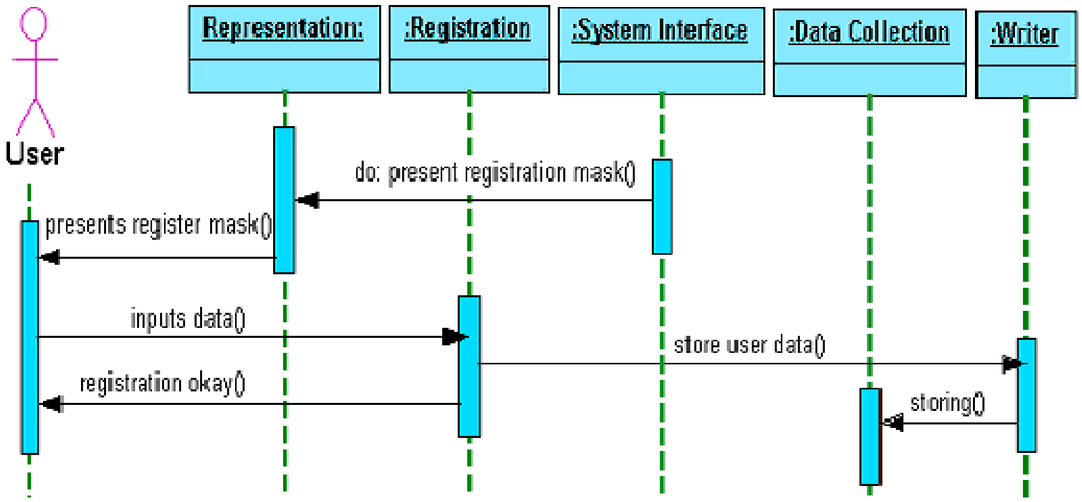


Figure 5.1: Sequence Diagram of Use Case Register

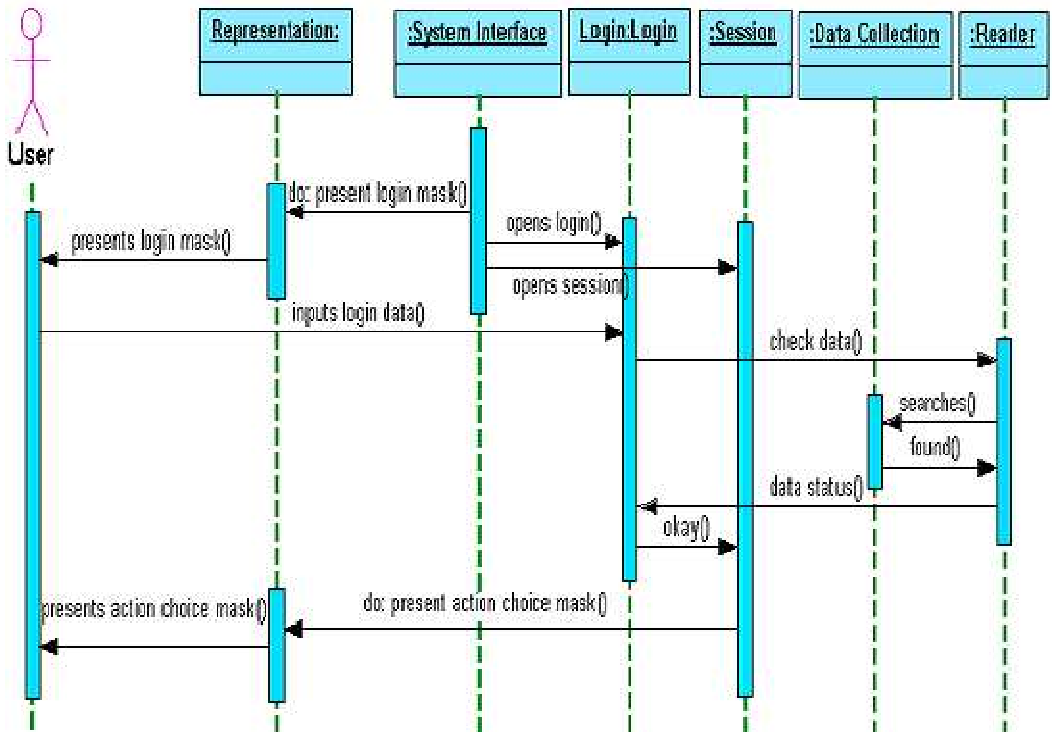


Figure 5.2: Sequence Diagram of Use Case Login and Action Choice

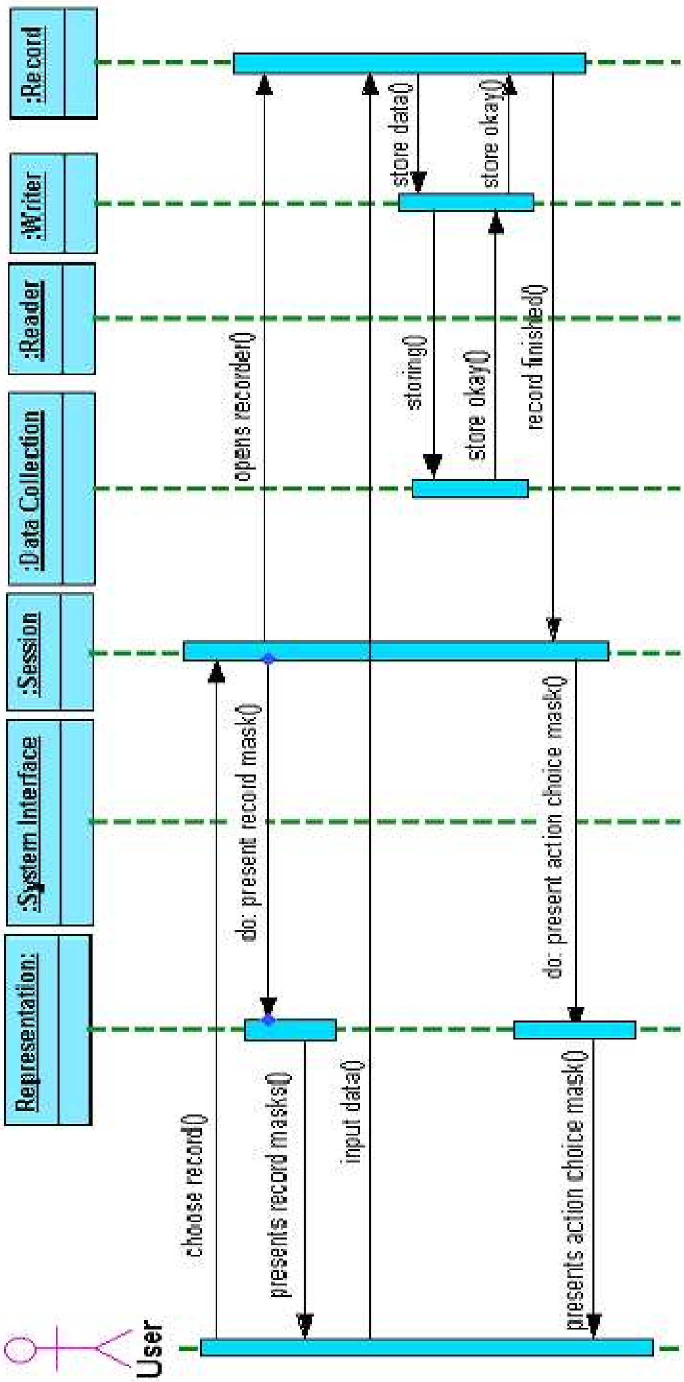


Figure 5.3: Sequence Diagram of Use Case Record

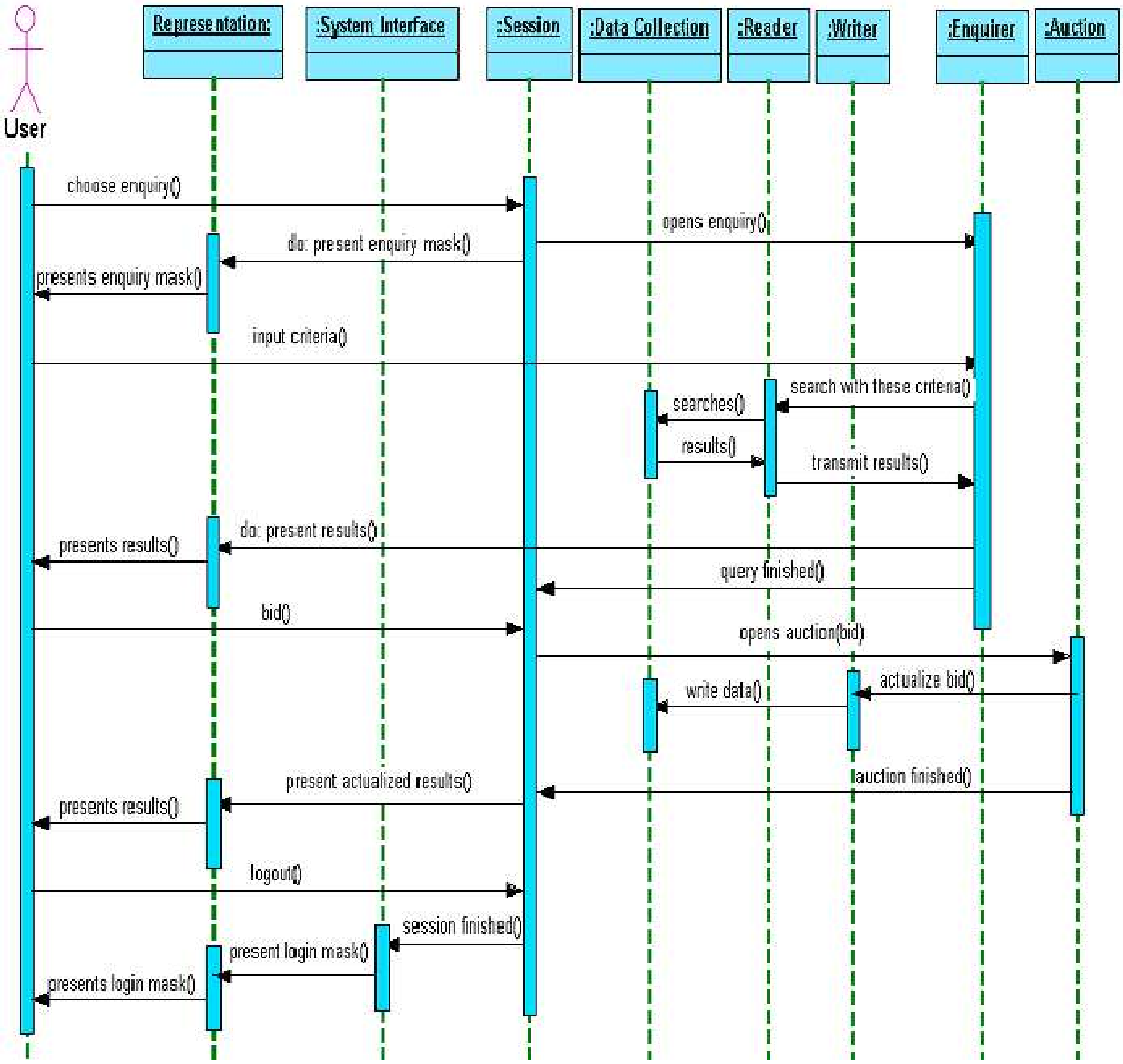


Figure 5.4: Sequence Diagram of Use Case Enquiry and Auction

Appendix A

Protocols

A.1 October 31, 2001

1. Person responsible for Protocoll
   Rescher Robert

2. Participating People
   Mag. Dipl. Ing. Schwaiger Roland, Rescher Robert, Strobl Elisabeth, Nidetzky Marion, Fimberger Lucia, Schobesberger Oskar

3. Topics talked about
   What is CRC?
   Presentation of our CRC cards

4. New Assignments
   To less classes

5. Open Questions
   Is the auction online?
   Can everybody participate the auction?
   Is there any restriction?
   Has the user to be identified?
   Can everyone record real-estates?

A.2 November 7, 2001

1. Person responsible for Protocoll
   Rescher Robert

2. Participating People
   Mag. Dipl. Ing. Schwaiger Roland, Rescher Robert, Strobl Elisabeth, Nidetzky Marion, Fimberger Lucia, Schobesberger Oskar

3. Topics talked about
   Presentation of the overworked CRC cards

4. New Assignments
   Auction is online.
   Everybody can paricipate an auction.
   The restriction is on the complete object via onion-modell.
   User has to be identified.
   Everyone, who wants to offer a real-estate can input, perhaps he has to pay for this.

A.3 November 14, 2001

1. Person responsible for Protocoll
   Rescher Robert

2. Participating People
   Mag. Dipl. Ing. Schwaiger Roland, Rescher Robert, Strobl Elisabeth, Nidetzky Marion, Fimberger Lucia

3. Topics talked about
   Presentation of the different use cases.
   Furtheron, a rough draft of user interface prototyping

4. New Assignments
   The auction is part of the system.

A.4 November 21, 2001

1. Person responsible for Protocoll
   Fimberger Lucia

2. Participating People
   Mag. Dipl. Ing. Schwaiger Roland, Rescher Robert, Fimberger Lucia, Schobesberger Oskar

3. Topics talked about
   Presentation of the class diagram.

4. New Assignments
   Do not think of implementation.
   Avoid inheritation.
   New classes: auction, session, login.
   User Interface Flow Diagram for WML and HTML and proof of functionality.

A.5 December 5, 2001

1. Person responsible for Protocoll
   Schobesberger Oskar

2. Participating People
   Mag. Dipl. Ing. Schwaiger Roland, Rescher Robert, Strobl Elisabeth, Fimberger Lucia, Schobesberger Oskar

3. Topics talked about
   Presentation of our user interface flow diagrams.

4. New Assignments
   Make HTML and XML site.
   Sequence diagram.
   Activity diagram.

A.6 December 12, 2001

1. Person responsible for Protocoll
   Rescher Robert

2. Participating People
   Mag. Dipl. Ing. Schwaiger Roland, Rescher Robert, Strobl Elisabeth, Nidetzky Marion, Fimberger Lucia, Schobesberger Oskar

3. Topics talked about
   Presentation of our activity and sequence diagrams.
   WML and HTML will be sent.